Intelligent Systems

Student no: 24678805

Coursework one

CALLUM MCLAUGHLIN

2021

Contents

[**Portfolio One** 2](#_Toc88143794)

[**Task 1.1** 2](#_Toc88143795)

[**Task 1.2** 2](#_Toc88143796)

[**Task 1.3** 3](#_Toc88143797)

[Instance 9 3](#_Toc88143798)

[Instance 10 3](#_Toc88143799)

[Instance 11 3](#_Toc88143800)

[**Portfolio Two** 4](#_Toc88143801)

[**Task 2.1** 4](#_Toc88143802)

[Confusion Matrices 4](#_Toc88143803)

[Recall 4](#_Toc88143804)

[Precision 4](#_Toc88143805)

[F-Measure 4](#_Toc88143806)

[K-NN Confusion Matrix 5](#_Toc88143807)

[Decision Tree Confusion Matrix 5](#_Toc88143808)

[K-NN Evaluation Metrics 6](#_Toc88143809)

[Decision Tree Evaluation Metrics 6](#_Toc88143810)

[Performance Analysis 7](#_Toc88143811)

[Discussion 7](#_Toc88143812)

[**Task 2.2** 7](#_Toc88143813)

[**Portfolio 4** 8](#_Toc88143814)

[**Task 4.1** 8](#_Toc88143815)

[**Task 4.2** 8](#_Toc88143816)

[Method Description 8](#_Toc88143817)

[Calculations 8](#_Toc88143818)

[**Task 4.3** 9](#_Toc88143819)

[References 10](#_Toc88143820)

# **Portfolio One**

## **Task 1.1**

The Iterative Dichotomiser 3 (ID3) algorithm is named as such because the algorithm iteratively (repeatedly) dichotomizes (divides) features into two or more groups at each step (Sakkaf, 2020). Invented by Ross Quinlan, the ID3 algorithm uses a top-down greedyapproach to build a decision tree. In simple words, the top-down approach means that we start building the tree from the top and the greedy approach means that at each iteration we choose the best feature at that point to create a node.

The advantages of ID3 are that it is understandable prediction rules are created from the training data. Finding leaf nodes enables test data to be pruned which also means that the number of tests is reduced. It builds a short tree in a short amount of time, and it only needs to test enough attributes until all data is classified (RapidMiner Studio Core, 2021).

The limitations of this algorithm are that data could be over-fitted or over-classified if only a small sample of data is tested and only one attribute can be tested at a time for deciding. ID3 can also become overly complex and confusing when used on a large-scale data set.

## **Task 1.2**

A picture containing text, sky, map, bunch

Description automatically generated

As seen in the diagram above, the first step was to choose a feature within the dataset of instances with the most common value. The feature that was chosen was Humidity as it had 5 ‘High’ values which was the most common value across all features. The next step was to split the instances into those that had normal humidity and those that had high humidity.

I have displayed this by writing the instance numbers that were split into each branch alongside the feature value. E.g., Normal humidity (5,6,7) means that instances 5,6,7 had a normal humidity value. Next, I went to the left with Normal Humidity and the most common value among those was outlook, both outlook instances had values of “Rainy” and one had overcast which could then be determined as “YES” as the only instance of overcast and normal humidity says “YES” in the training data. The outlook value “sunny” could also be determined as “YES” as there are no more instances to continue the tree. For the value “Rainy” in outlook, the value “Windy” was the most common and was chosen for the next value in the tree. Windy only had two instances to decide, one of which had a true value, and one had a false value, meaning “NO” and “YES” were assigned for these instances, respectively.

This was repeated for each group on instances and whenever all instances matched the same outcome, e.g., Humidity - Normal Humidity (5,6,7) – Outlook – Overcast (7) – YES, because all instances that had normal humidity and overcast gave the ‘YES’ decision, we can do the same. This same process continues until all nodes are displayed and all decisions shown.

## **Task 1.3**

### Instance 9

We start at Humidity at the top of the tree, the value for this is Normal so we go to the left branch which then activates the outlook node, the value for which is sunny which leads to the decision of **YES.**

### Instance 10

Starting at Humidity, the value is also normal, so we go to the left branch of the tree again and activate the outlook node which has a value of Rainy in this instance, so we go to the left branch of outlook which activates the Windy node which has a value of False in this instance which leads to the decision **YES.**

### Instance 11

We start again at the top from humidity which is Normal again and we go left once more which leads to activating the outlook node and the value for this instance is sunny which will lead to a decision of **YES.**

# **Portfolio Two**

## **Task 2.1**

## Confusion Matrices

A confusion matrix is a performance measurement for machine learning classification where the output can be two or more classes. Usually, it is displayed in a table with 4 different combinations of predicted and actual values. The possible results within a confusion matrix are True Positive (Predicted positive and its true), True Negative (Predicted Negative and its true), False Positive (Predicted Positive and its false) and False negative (Predicted negative and its false) (Narkhede, 2018) .

Confusion matrices are also extremely useful for measuring Recall, Precision and F-Measure.

### Recall

Recall is a metric that quantifies how many correct positive predictions were made from all the positive predictions that could have been made (Brownlee, 2020).

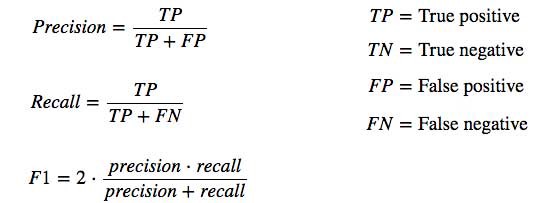
### Precision

Precision quantifies the number of positive class predictions that belong to the positive class. (Brownlee, 2020).

### F-Measure

F-Measure provides a single score that is a combination score of precision and recall concerns into one number. (Brownlee, 2020)

The formulas shown below show how Recall, Precision and F-Measure can be calculated using the results from confusion matrices.



Formula’s reference (Hui, 2018)

### K-NN Confusion Matrix

|  |  |  |  |
| --- | --- | --- | --- |
| **Actual Events** | **Predicted Events** | | |
|  | **+** | **-** |
| **+** | **TP = 4** | **FN = 0** |
| **-** | **FP = 5** | **TN = 1** |

### Decision Tree Confusion Matrix

|  |  |  |  |
| --- | --- | --- | --- |
| **Actual Events** | **Predicted Events** | | |
|  | **+** | **-** |
| **+** | **TP = 3** | **FN = 1** |
| **-** | **FP = 0** | **TN = 6** |

### K-NN Evaluation Metrics

#### Precision

A picture containing chart

Description automatically generated

#### Recall

A screenshot of a computer

Description automatically generated with low confidence

#### F – Measure

Diagram

Description automatically generated

### Decision Tree Evaluation Metrics

#### Precision

A screenshot of a computer

Description automatically generated with low confidence

#### Recall

A picture containing shoji, clock

Description automatically generated

#### F – Measure

A picture containing diagram

Description automatically generated

### Performance Analysis

It is clear from the results that the decision tree algorithm presents the most accurate results. The decision tree algorithm had 3 true positives and 6 true negatives leaving only one false negative and not a single false positive which is what is important in this specific case as false positives in a cancer detection algorithm could have profound consequences for that patient’s treatment. whereas the K-NN algorithm had 4 true positives and 1 true negative with the other half (5) of the instances being predicted as false positives. The decision tree algorithm presents perfect precision (1) and a superior F-measure (2.625) in comparison to the K-NN algorithm which has only 0.444 precision and an F-Measure of 1.444 although it does have a perfect recall score where the decision tree does not.

### Discussion

When trying to decide which machine learning algorithm to use for the given task you must first categorize the problem. If it is labelled data, then it is a supervised learning problem. If it is unlabelled data and the purpose is to find structure, it is an unsupervised learning problem. If the task requires the algorithm to optimise an objective function by interacting with an environment, then it must be a reinforcement learning problem (Almaliki, 2019).

The next step is to understand your data as it is the raw material in the entire process. Understanding the data plays a huge role in the success and accuracy of the algorithm. Some algorithms work well with large data sets whereas others can only work with small sets or categorical data or numerical input.

Next you would find available algorithms after you have categorized the problems and you understand your data. Ann algorithm must be identified that is applicable and practical to implement within a reasonable period. After choosing the algorithm it would then be implemented and optimised accordingly.

## **Task 2.2**

The best algorithm for cancer detection would be the K-NN algorithm as it predicted every patient with cancer correctly whereas the decision tree algorithm had one false negative which is exactly what the doctors are concerned about. The K-NN algorithm did have a perfect recall score (1) in comparison to the inferior score of the Decision Tree Algorithm (0.75) which would also be an argument for using this algorithm as it predicted every single patient that has cancer correctly and never gave a false negative. The K-NN algorithm is the best choice for the hospital as it gets the results, they need by being able to predict every patient that has cancer correctly and not give any false negatives which is exactly what the doctors wanted.

# **Portfolio 4**

## **Task 4.1**

The Naïve bayes algorithm is a classification technique which is based on the Bayes’ Theorem with the assumption of independence among predictors (Ray, 2017). In simple terms, the classifier assumes that the presence of a specific feature in a class is unrelated to the presence of any other feature.

The advantages of naïve bayes include the fact that it is simple and easy to implement, it does not require lots of training data, it can handle both discrete and continuous data and it is fast and can be used to make real-time predictions.

The limitations of this algorithm are that Naïve Bayes assumes that all features are independent, which is rarely the case. This limits the applicability of this algorithm to real world use. The algorithm also faces the ‘Zero-Frequency problem’ where it assigns zero probability to a categorical variable whose category in the test data set was not available in the training dataset. Its estimations can also be wrong in some cases so the probability outputs cannot be taken very seriously (Shah, 2021) .

## **Task 4.2**

**To ensure that I remember the value I am dealing with and for ease of reading, instead of just using 1 or 0 I have decided to use the following:**

**Survived (1) = 1**

**Survived (0) = 0**

### Method Description

The first step was then to compute the probability of each label which can be seen as the first 2 calculations in the Calculations section below. Next, I calculated the probability of each feature and each value of that feature which has been displayed in the 3 tables below, one for each feature type. Next, I calculated the probability of Instance 5 which can be seen in Task 4.3.

### Calculations

Probability of Survived (1) = P (Survived (1)) = ¼ = 0.25

Probability of Survived (0) = P (Survived (0)) = ¾ = 0.75

|  |  |  |
| --- | --- | --- |
| Feature: Ticket Class | Survived 1 | Survived 0 |
| First | ½ = 0.5 | ½ = 0.5 |
| Second | 0/1 = 0 | 1/1 = 1 |
| Third | 0/1 = 0 | 1/1 = 1 |
| Feature: Age | Survived 1 | Survived 0 |
| Young | 1/1 = 1 | 2/3 = 0.66 |
| Old | 0/1 = 0 | 1/3 = 0.33 |

|  |  |  |
| --- | --- | --- |
| Feature: Cabin | Survived 1 | Survived 0 |
| Yes | 1/1 = 1 | 2/3 = 0.66 |
| No | 0/1 = 0 | 1/3 = 0.33 |

## **Task 4.3**

P (Survived (1) | third old no) = P (Survived (1)) \*P (third | Survived (1) \*P (old | Survived (1)) \*P (no | Survived (1) = 0.25 x 0 x 0 x 0 = 0

P (Survived (0) | third old no) = P (Survived (0)) \*P (third | Survived (0) \*P (old | Survived (0)) \*P (no | Survived (0) = 0.75 x 0.33 x 0.33 x 0.33 = 0.02695

P (Survived (0) | third old no) **>** P (Survived (1) | third old no)

**Instance 5 Predicted Label = Survived (0)**

# References

Almaliki, Z. A. (2019, March 19). *Do you know how to choose the right machine learning algorithm among 7 different types?* Retrieved from Towards Data Science: https://towardsdatascience.com/do-you-know-how-to-choose-the-right-machine-learning-algorithm-among-7-different-types-295d0b0c7f60

Brownlee, J. (2020, 01 03). *How to Calculate Precision, Recall, and F-Measure for Imbalanced Classification*. Retrieved from Machine Learning Mastery: https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/

Hui, J. (2018, 03 07). *mAP (mean Average Precision) for Object Detection*. Retrieved from Jonathan Hui: https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173

Narkhede, S. (2018, 05 09). *Understanding Confusion Matrix*. Retrieved from Towards Data Science: https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

RapidMiner Studio Core. (2021, 11 01). *ID3*. Retrieved from Rapid Miner: https://docs.rapidminer.com/latest/studio/operators/modeling/predictive/trees/id3.html

Ray, S. (2017, September 11). *6 Easy Steps to Learn Naive Bayes Algorithm with codes in Python and R*. Retrieved from Analytics Vidhya: https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/#:~:text=Naive%20Bayes%20Model-,What%20is%20Naive%20Bayes%20algorithm%3F,presence%20of%20any%20other%20feature.

Sakkaf, Y. (2020, 03 31). *Decision Trees: ID3 Algorithm Explained*. Retrieved from Towards Data Science: https://towardsdatascience.com/decision-trees-for-classification-id3-algorithm-explained-89df76e72df1

Shah, R. (2021, 03 11). *Naïve Bayes Algorithm's Advantages and Disadvantages*. Retrieved from Kaggle: https://www.kaggle.com/getting-started/225022